AFRL-HE-BR-TR-2005-0162

Air Force Research Laboratory

A VERIFICATION AND ANALYSIS OF THE USAF/DOD FATIGUE MODEL AND FATIGUE MANAGEMENT TECHNOLOGY

Scott Chaiken

HUMAN EFFECTIVENESS DIRECTORATE
BIOSCIENCES AND PROTECTION DIVISION
FATIGUE COUNTERMEASURES BRANCH
2485 GILLINGHAM DRIVE
BROOKS CITY-BASE TX 78235-5105

November 2005

NOTICES

This report is published in the interest of scientific and technical information exchange and does not constitute approval or disapproval of its ideas or findings.

This report is published as received and has not been edited by the publication staff of the Air Force Research Laboratory.

Using Government drawings, specifications, or other data included in this document for any purpose other than Government-related procurement does not in any way obligate the US Government. The fact that the Government formulated or supplied the drawings, specifications, or other data, does not license the holder or any other person or corporation, or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

The Office of Public Affairs has reviewed this paper, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This report has been reviewed and is approved for publication.

//SIGNED//
SCOTT R. CHAIKEN
Project Scientist

//SIGNED//
F. WESLEY BAUMGARDNER, Ph.D.
Deputy, Biosciences and Protection Division

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
November 2005

2. REPORT TYPE
Interim

3. DATES COVERED *(From - To)*
01 Jan 2005 to 30 Sept 2005

4. TITLE AND SUBTITLE
A Verification and Analysis of the USAF/DOD Fatigue Model and Fatigue Management Technology

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62202F

6. AUTHOR(S)
Scott Chaiken

5d. PROJECT NUMBER
7757

5e. TASK NUMBER
P9

5f. WORK UNIT NUMBER
05

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Human Effectiveness Directorate
Biosciences and Protection Division
Fatigue Countermeasures Branch
2485 Gillingham Drive
Brooks City-Base, TX 78235

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Human Effectiveness Directorate
Biodynamics and Protection Division
Fatigue Countermeasures Branch
2485 Gillingham Drive
Brooks City-Base, TX 78235

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/HE

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-HE-BR-TR-2005-0162

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
We verified the Windows® software application of the Sleep, Activity, Fatigue, and Task Effectiveness (SAFTE) applied model. The application, the Fatigue Avoidance Scheduling Tool (FASTTM) was re-engineered as a clone from the SAFTE specification. The verification considered nine sleep/wake schedules that were representative of applications of the fatigue model. Of the nine, eight were considered fully verified in the sense that the clone's output matched FAST's output well, or the differences in the clone and FAST outputs could be understood as an implementation choice. The ninth schedule was very nearly verified and seemed to require only a minor adjustment to FAST for full verification. One of the benefits of the verification process was an analysis of how FAST implementation choices impacted predictions relative to simpler implementation choices made by the clone. Most of these implementation differences lead to inconsequential prediction differences; however, the difference highlighted by schedule 7 bears further consideration.

15. SUBJECT TERMS
Fatigue Management Tools, Fatigue Management, Fatigue

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Scott Chaiken
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	Unclass	46	**19b. TELEPHONE NUMBER** *(include area code)*
Unclass	Unclass	Unclass			(210) 536-2870

CONTENTS

SUMMARY

We verified the Windows® software application of the Sleep, Activity, Fatigue, and Task Effectiveness (SAFTE) applied model. The application, the Fatigue Avoidance Scheduling Tool (***FAST***™) was re-engineered as a clone from the SAFTE specification. The verification considered nine sleep/wake schedules that were representative of applications of the fatigue model. Of the nine, eight were considered fully verified in the sense that the clone's output matched FAST's output well, or the differences in the clone and FAST outputs could be understood as an implementation choice. The ninth schedule was very nearly verified and seemed to require only a minor adjustment to FAST for full verification. One of the benefits of the verification process was an analysis of how FAST implementation choices impacted predictions relative to simpler implementation choices made by the clone. Most of these implementation differences lead to inconsequential prediction differences; however, the difference highlighted by schedule 7 bears further consideration.

PREFACE

This report covers the project period of 1 January to 30 September 2005. The work was performed under AFRL Job Order Number 7757P905.

The project manager was Dr. James C. Miller, Fatigue Countermeasures Branch (AFRL/HEPF), Biosciences and Protection Division, Air Force Research Laboratory, Brooks City-Base TX. Dr. Miller also served as reviewer and editor for the report.

INTRODUCTION

The Sleep, Activity, Fatigue, and Task Effectiveness (SAFTE) applied model, or simulation, is a consensus model for the effects of fatigue on "cognitive effectiveness", drawing from the wisdom derived from empirical studies of both normal and abnormal (for example, restricted) sleeping conditions. The SAFTE model describes how effectiveness depletes with sustained wake and replenishes with sleep. It uses a simple but highly dynamic, mathematical model, viewing the fatigue process in terms of a "reservoir" analogy. The reservoir leaks during the waking hours and replenishes during sleeping hours. The dynamics of the model are defined by the wake/sleep history of the person modeled. Because the input to the model is (conceptually) just a Boolean array of minutes (for example, true if sleeping, false if awake) plus relevant context information (for example, time-zone shifts, duty locations), predictions for very complicated schedules are possible. For a more detailed specification of the SAFTE model, including additional background, see Hursh et al. (2004) and the patent site[1].

The output of the SAFTE model is a cognitive effectiveness prediction, which is usually just referred to as "effectiveness" for brevity. Effectiveness is a uni-dimensional construct scaled from about 0 to about 100% (*i.e.*, values greater than 100% are possible). This construct *ideally* represents *performance quality*. "Ideally" is italicized to stress that "performance quality" is not equivalent to "cognitive effectiveness". Observed task performance functions can be different from the expected cognitive effectiveness functions (*for example*, consider a task that can be done acceptably well at both an 80% and a 100% level of cognitive effectiveness). SAFTE does

1. The following links to patent web-site: http://appft1.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PG01&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.html&r=1&f=G&l=1&s1='20030018242'.PGNR.&OS=DN/20030018242&RS=DN/20030018242

not model the extent to which a specific task is susceptible to fatigue, but only models a task-independent resource level, as it varies with sleep and wake, in the individual. Every fatigue model we know of has this characteristic, so it is unfair to cite that particular characteristic as a shortcoming, *per se*. Even so, one can still say that SAFTE (and models like it) predict task performance accurately given that the task in question depends on "cognitive effectiveness" (whatever that turns out to be).

The "Fatigue Avoidance Scheduling Tool (FAST) is Windows® software that provides a user interface for the SAFTE model, meaning FAST allows easy input of a proposed work schedule (*i.e.*, sleep and wake intervals) and reports SAFTE model predictions for that schedule. FAST displays "effectiveness" plotted against time into the schedule. Hence, schedulers, planners and mission commanders can look at proposed schedules, minute by minute, from the perspective of the most scientifically up to date model of fatigue.

Both SAFTE and FAST are patented by SAIC (Science Applications International Corporation) and NTI (Nova Technology, Incorporated), though both have been developed under DoD funding (Small Business Innovative Research and Cooperative Research and Development Agreements). Much of the recent support for SAFTE/FAST development has been through the Warfighter Fatigue Countermeasures Branch of the Air Force Research Laboratory (AFRL/HEPF).

Purpose of this Paper

SAFTE/FAST has begun the USAF Verification, Validation and Accreditation (VV&A) process for software. Accreditation "is the formal certification that a model or simulation is acceptable for use for a specific purpose" (Defense Acquisition University, 2004). A product generally gets

accredited after the Verification and Validation requirements are met. Validation "is the process of determining the manner and degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model, and of establishing the level of confidence that should be placed on this assessment" (Defense Acquisition University, 2004) . In our particular case, validation is handled by reference to the scientific literature on sleep and fatigue that SAFTE mimics, explains and predicts (reviewed in Hursh et al., 2004).

Verification is most germane to this paper. Verification "is the process of determining that *a model implementation* accurately represents the *developer's conceptual description and the specifications* to which the model was designed" (Defense Acquisition University, 2004; italics added). In this particular case, we are interested in how accurately and comprehensively the FAST program implements what we understand to be the "intentions" and mechanics of the SAFTE model.

To characterize the *verification* component of VV&A merely as "quality assurance" for software greatly underestimates the usefulness of the verification procedure. Although some reality checks of the software are required formally to expedite the VV&A process for FAST, verification also increases our (i.e. the DoD's) basic-level understanding of SAFTE as a scientific theory of fatigue effects. That is, the procedure of verification allows us to more fully appreciate the limitations of the model and identifies the areas where it could be better specified.

One thing to keep in mind about our "verification" effort is that we are not an independent evaluating organization, and therefore can not technically fulfill the verification requirement. However, we are an interested party in the technology in the sense that we want the fatigue model to be the best model possible and can provide evidence that FAST would be well verified by others should that be deemed necessary. This report is intended as a proactive step in that

direction. If the Air Force wishes to *independently* verify FAST via an external organization, then this report should make that process easier.

METHOD

Our verification is limited to the most important moderators of effectiveness, namely the reservoir and circadian dynamics (these are also the most accessible aspects of the published SAFTE specification in Hursh *et al.*, 2004). A complete verification of all the intricacies of the SAFTE model (and FAST's implementations of them) is beyond the scope of this paper. Fine-grain validations and verifications (for example concerning light, drug, and time-zone-shifting effects) should be accomplished by the future stake-holders in this technology.

Our basic method for SAFTE to FAST verification is "reverse-engineering." We take the most detailed specification of the SAFTE model in the open literature (Hursh et. al., 2004) and construct a parallel FAST clone (hereafter "clone"). If we find that our clone reproduces FAST output, we have verified FAST and have simultaneously shown that SAFTE has been specified with enough detail to be considered a public theory of fatigue, sleep and activity (albeit a patented theory).

Reverse engineering is not our only method. Some of the verification process is accomplished via the breadth and choice of work schedules we explore. These work schedules are motivated by what has been reported in the scientific literature and by our likely future applications for FAST. Independently of the output produced by the clone, we can assess whether FAST acts as we expect. If FAST surprises us, we first determine if the effect is derivable from the SAFTE specification. If it is not, then this is a verification issue to be resolved. However, if we do find out surprising effects are derivable from SAFTE, with a clone in hand, we may gain additional insight into those effects. Both outcomes are valuable (and both outcomes were obtained in this

effort).

PROCEDURES

Equipment and Functional Analysis

We used the Java 5.0 Platform Standard Edition (www.java.sun.com) as the development language, while code integration and compiling was managed by the Netbeans Java IDE version 3.6 (Integrated [software] Development Environment - the latest version of this IDE (4.1) is available at www.netbeans.org). Both of these software engineering tools are available for free from their respective domains. All the programming was done by the author.

Software production in the DoD is expensive, even if the DoD agency does much of the actual programming "in-house," as is the case here for the clone. The engineering cost of the clone software was minimized by using open-source software available at the time the project started (roughly a year prior to this writing). An analysis of the functions required for a FAST clone identified three broad needs or functions: (1) a schedule-input part, providing a description of a schedule to be used as an input for SAFTE modeling, (2) an implementation of the SAFTE model, and (3) an output display part, for graphically displaying SAFTE predictions.

Open-source, easily-modified solutions could be found for functions 1 and 3. In particular, schedule inputting, function 1, was accomplished via a straightforward XML description of a schedule combined with the Simple API for XML (SAX) processing framework. The SAX framework has been illustrated in a pedagogical program, published by David Matuzsek

(http://www.cis.upenn.edu/~matuszek/cit597-2002/Examples/second-sax-example.html) for educational use. This program was extremely helpful as a starting position for function 1.[2]

For function 3, prediction display, we leveraged JFreeChart, which can be found at http://www.jfree.org/jfreechart/index.php . This is a free general-purpose chart/figure library under the LGPL open-source license (http://www.gnu.org/licenses/lgpl.html). This is a rather extensive library with many components. However, one particular self-contained example/demo org.jfree.chart.demo.XYLineAndShapeRendererDemo.java served as a useful starting position and effectively summarizes the components we leveraged.

The SAFTE model implementation (function 2) also required some textual data outputting. These data not only reported program state information (to provide checks that the program was acting as expected) but also provided data entry conveniences. While JFreeChart displays "tooltip" access to needed data, these are not amenable to cutting and pasting into results tables. Otherwise, the SAFTE model implementation (discussed more below), constituted the bulk of function 2 and was the best module upon which to concentrate the in-house effort, from the point of view of understanding and investigating the dynamics of the SAFTE model.

Reference Materials

After software functional baselines were established, the reverse-engineering project started in earnest. This required a working copy of the FAST program (we started with version 1.0.23 and ended with version 1.0.26) along with a careful reading and decomposition of Hursh *et. al.*, 2004. We also used the patent website, although we came to realize somewhat late in the process of reverse engineering that this was available as a resource . This turned out to be a fortuitous

2. The XML schema for schedule input is explained as an extended comment (not as a DTD) in baseline_schedule.xml provided in the xml_files directory of the final archive of this project.

oversight in the sense that without the pseudo-code present at the patent site, we ended up investigating some simpler implementation choices than are specified by the patent (although we feel our implementation choices are still covered by the patent). This allowed us to make some assessment of whether schedule predictions, *depended* upon implementation choices. We think it is important (or at least useful) to keep track of whether an implementation choice matters to prediction, especially if such choices do not reflect meaningful theoretical differences. During the reverse-engineering period, we also gained insight from several email exchanges with the NTI and SAIC principals, Drs. Steven Hursh and Douglas Eddy.

Overview of Clone Code

The clone is an object-oriented perspective of the model (a necessity if you are working with Java). This perspective breaks down the functionality of the SAFTE model in a fashion similar to the breakdown of the global FAST functionality described above. A *ModelImpl* class (a "model implemented" class) orchestrates the various sleep theory components from within a schedule loop that updates states of the sleep objects at minute intervals. *ModelImpl* also bridges the FAST functions (1 and 3) to the SAFTE model implementation, converting the XML representation of a work schedule into an array of booleans (true for sleep; false for awake) and conveying the resultant *integer* effectiveness predictions (i.e. a 3-decimal percentage times a 1000) to JFreeChart components for plotting.

What "sleep theory components" does the *ModelImpl* class orchestrate? These are other classes in the "sleep", "inertia", "awake", and "circadian" software *packages*. Packages are simply a file-directory structure underneath the clone's main package, which contains *ModelImpl*. Packages provide a conceptual expansion space for further class differentiation relevant to the package; however, currently each (SAFTE-inspired) package has only one Java class in it. The class in a package "chunks" the SAFTE mathematics deemed most appropriate to that package

name. For instance, the "sleep" package holds a *SleepIntensity* class, the "awake" package holds a *PerfUse* class (a "performance use" class), the circadian package holds a *CircadianRhythm* class, and so on. Each Sleep component class contains methods deemed relevant to the sleep component. For instance, the *SleepIntensity* class has a *getSleepDebt(), getSleepPropensity(),* and *getCurrentResevoirCapacity()* method. These map directly into SAFTE calculations. Finally, objects "talk" to other objects as needed to express the model (for instance, the getSleepPropensity() method needs to receive information from the CircadianRhythm object, as sleep propensity varies as a function of circadian rhythm).

We made no theoretical statements to SAFTE by using an object-orientation framework. We use it because it is fundamental to Java and other programming languages. However, we believe object-orientation can greatly improve the readability of programs. Less readable code may have been created without that framework. Modern IDEs such as NetBeans afford greater readability of the code, and may be a more fluent medium within which to view a mathematical model (than say a research paper). With an IDE one can navigate the package structure, their classes and the methods within the classes rapidly. Thus, the code browser becomes a knowledge browser, to some extent, depending on whether class, method and state variable names are chosen with enough care to be "self-documenting". The self-documenting ideal was followed to the extent that time allowed.

RESULTS

In summary, nine schedules were considered. We obtained close agreement between the clone and FAST on the first 6 schedules. For the seventh and eighth schedules there were substantial disagreements. Upon analyzing the disagreements, we concluded that the differences were due to the clone deviating from the actual SAFTE specification as detailed at the patent web site. However, because the implementation differences *do not* map into a *theoretical position* (*i.e.*,

neither implementation is technically "right" or "wrong"), we feel it's more useful to keep the clone different from FAST. The ninth schedule also led to a disagreement that appeared to be an anomaly, given that we could not derive FAST's output from our understanding of the SAFTE specification. However, we believe this last disagreement reflects only an easily-corrected programming bug.

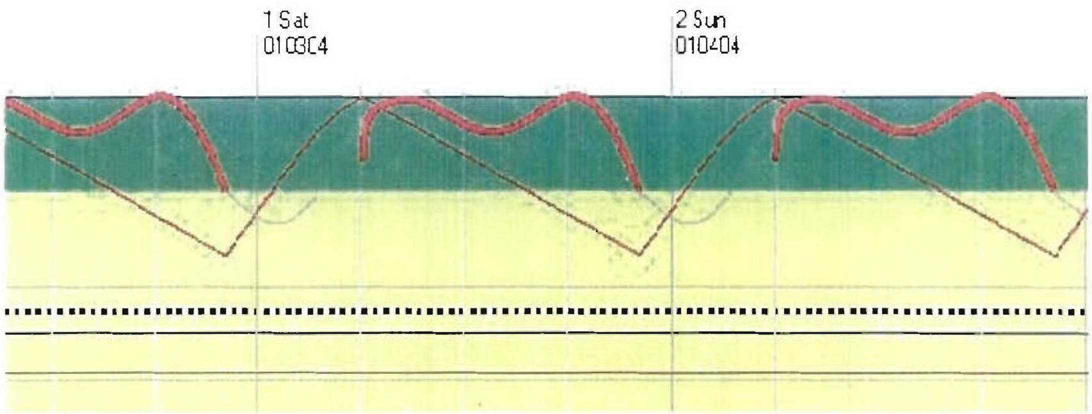
Detailed Schedule Reviews

Tables presenting the results for each schedule are presented as separate "tables". The schema for the tables can be discussed in the context of Table 1. Each "table" is headed by a short abstract of the schedule predicted (e.g. what it is with optional comments about what was found). This is followed by a two-part figure showing FAST graphical output and then the clone's output. These graphics are followed by an *optional* table containing numeric comparisons between FAST and clone. The table is optional, in the sense that if FAST and clone show obvious disagreements in the graphics, it makes no sense to show the same with numbers. If tabled numbers are present, it's because the similarities between FAST and clone are qualitatively good, making closer inspection worthwhile.

The tabled values show both the "Effectiveness" and the "Reservoir Level" predictions for each implementation. These are provided for representative times in the schedule, where "Date/Time" and "Events" columns describe the times. Effectiveness is determined mainly by the reservoir level; however, it is also determined by the phase of the circadian rhythm. Showing the reservoir level, alone, allows a look at a purer aspect of SAFTE model output, and this is sometimes useful for diagnosing FAST/clone differences.

Table 1: baseline schedule

Baseline schedule 2200-0600 sleep-wake times and a starting acrophase of 1700.



		Effectiveness		Resevoir Level	
Date/Time	**Event**	**FAST**	**Clone**	**FAST**	**Clone**
Jan04_0600	wake	982	981	2880	2880
Jan04_0601	wakeInertia	935	935	2880	2879
Jan04_2200	sleep	902	902	2400	2400
Jan04_0830	firstPeak	997	997	2804	2805
Jan04_1815	secondPeak	100	100	2512	2512
Jan05_0100	lowestTrough	866	867	2616	2615

Table 1 shows the "baseline" schedule (sleep/wake 2200/0600) at asymptote. In the FAST figure the saw-tooth function is showing reservoir levels, while the sinusoidal function is showing

effectiveness levels. The latter function is in fact the circadian rhythm added to the reservoir level function. The circadian pattern is thought to compensate for the reservoir decline during the course of the day, the goal being to keep effectiveness at or above the 90% level (the green band in the FAST figure). To do this, the circadian rhythm peak time is set in SAFTE to occur three hours after the midpoint of the awake period, which as a rule of thumb reflects the optimal placement of the circadian rhythm and is also necessary to fit the empirical data. This way of defining where the circadian peak is anchored also defines a target circadian position toward which to move if the sleep/wake schedule is shifted in time. The current, highest level of the circadian oscillation (*i.e.*, after circadian shifting has stopped), or a *new* target position (*i.e.*, resulting from a schedule change that initiates a circadian shift), is referred to as the "acrophase" in the table captions and the text below.

The waking period for FAST is generally indicated in FAST by a bold red line on the effectiveness curve or where the reservoir function has a negative slope. Only effectiveness is plotted in the clone figures. In the clone figures, the start of an awake period is indicated by a green bullet on the (thin red) effectiveness function, after which, up until the next blue bullet, the individual is awake. Note that the initial period of waking, just after the green bullet, shows effectiveness spiking downward (also seen in FAST by the separation of the light grey and red function segments). This indicates the presence of "sleep inertia", an upto 5% drop in effectiveness that occurs upon awakening.[3] The beginning of a sleep period is indicated by a blue bullet, after which, up until the next green bullet, the individual is sleeping.

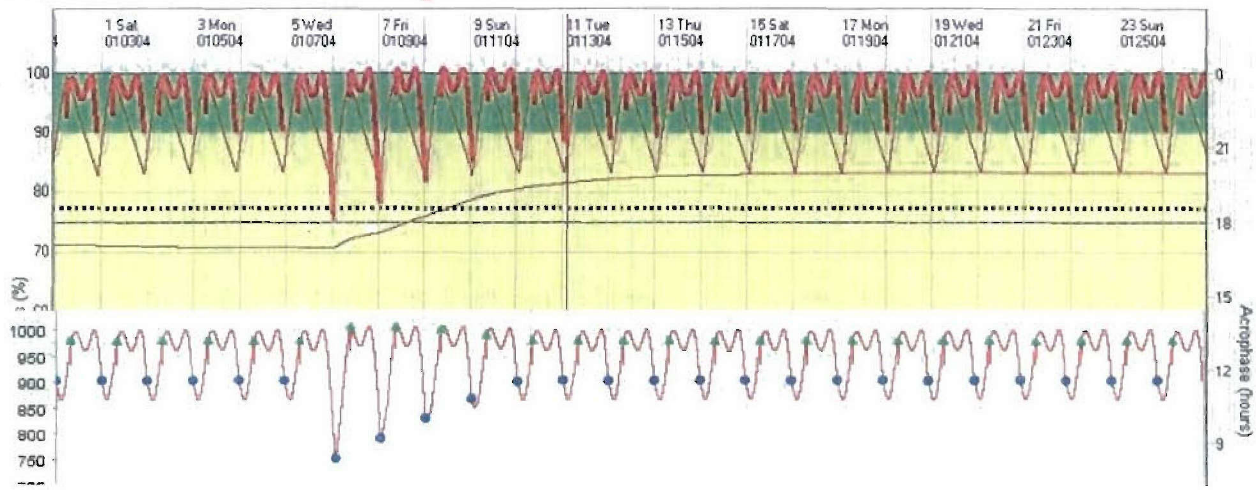
---

3. For the "waking bullet" only, the clone expresses the effectiveness level *without considering inertia*; this allows the waking bullet to precede the inertia spike, which looks better. In FAST, effectiveness is *always* expressed for the moment *starting* with the time requested. Therefore, for "wake" events *only*, one should compare FAST effectiveness *at the end of sleeping* (*e.g.*, 0659) to the clone at the beginning of waking (e.g. 0700). Using this approximation means the FAST/clone comparison at (3. continues) waking is off by 0.5 reservoir units (*i.e.*, the clone has leaked one more minute than FAST at this point).

Table 1 shows close agreement between FAST and clone on the baseline schedule. Peaks and troughs occur at the same place and have highly similar effectiveness and reservoir-level values. The remaining schedules show changes from this baseline and how FAST and clone adjust to the changes.

Table 2: 3hours_plus shift

Baseline schedule migrates to 0100-900 sleep-wake times, leading to a POSITIVE 3 hour acrophase shift.



		Effectiveness		Resevoir Level	
Date/Time	**Event**	**FAST**	**Clone**	**FAST**	**Clone**
Jan09_0100	sleep_CircShift_starts	751	752	2310	2310
Jan09_0900	wake	1006	1006	2825	2825
Jan09_0901	wakeInertia	963	963	2824	2825
Jan11_0100	sleep	852	830	2386	2371
Jan11_0900	wake	999	1003	2878	2871
Jan11_0901	wakeInertia	955	960	2878	2870
Jan13_0100	sleep	884	900	2400	2400
Jan13_0236	circShift_ends	864	877	2522	2511
Jan13_0900	wake	989	981	2880	2880

Table 2 is a 3-hour positive shift in bed time (sleeping from 0100 to 0900). This delay causes a 3-hour delay in where the circadian peak time should be positioned to reflect optimal placement. For this kind of shift, 3 days is typically needed to re-align the circadian rhythm to its optimal position (i.e., a positive shift, or phase delay, shifts at 1 day for each hour, Hursh et al., 2004). However, the shift rate for this schedule is actually 1.4 days/hour, owing to another factor: sleeping-in-the-daytime.

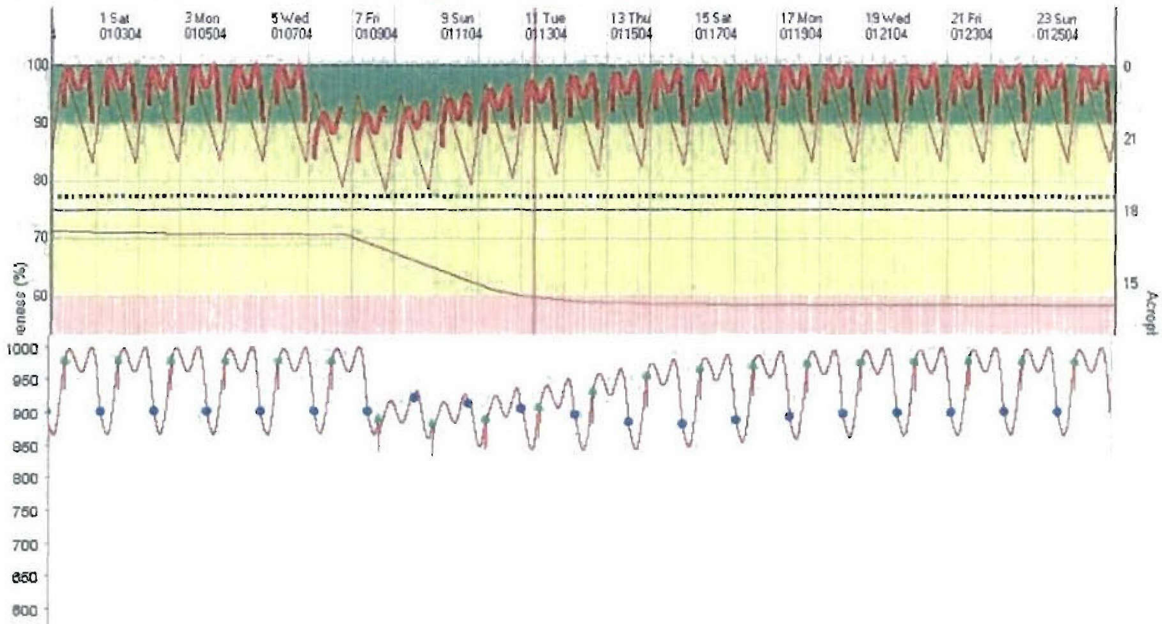
According to SAFTE, when one has to sleep in the daytime the cues that govern circadian shifting are less optimal, so shifting is slowed relative to its rule-of-thumb rate. According to the schedule and the clone's implementation, the hours from 0700 to 0900 are daytime hours, and FAST makes a similar assumption. In FAST the shifting dynamics are slowed by a 4-point scale that maps percentage bands for sleeping-in-the-daytime, [0-.16, .16-0.33, .33-.50, .50-1.0], onto shift-rate adjustment factors [1, 1.5, 2.0, 2.6]. For this particular schedule, assuming 25% of the sleep in the daytime, FAST predicts a shift rate of 1.5 days per hour (see patent point [0178]). The clone bases its shift-rate adjustment from "poor cues" by using a continuous formula: shiftRate = (shiftRate) * (1 + 1.6*p), where p is percentage slept in the daytime.[4] This produces values in the range 1 to 2.6 times the shiftRate, as p goes from 0 to 1.0. With this formula, and p = 25%, the clone predicts a shift rate of 1.4 day per hour of needed shift, which is similar to but not exactly the same as FAST.

4. This implementation difference occurred from failing to follow the patent specification early in the re-engineering process. As the clone's implementation seems a fair way to do it, and leads to only minimal deviation from FAST, we let the deviation continue to exist in the clone. However, at the 50% sleep-at-day point, FAST jumps to its maximum value, while clone continues to climb at a linear rate to the maximum value. Therefore, this implementation difference could matter for a schedule which just passes the 50% sleep-at-day point.

The numbers in the table are close, but there are slight discrepancies owing to the different shift-rates FAST and clone provide (1.5 *vs.* 1.4, respectively). However, using a FAST-provided shift-rate to produce clone outputs, the values would be closer. For instance, Jan11_0100 would be changed to 861 and 2391 for effectiveness and reservoir level, respectively, which are closer to the FAST values, 852 and 2386.

Table 3: 3hours_negative shift

Baseline schedule migrates to 1900-0300 sleep-wake times, leading to a NEGATIVE 3 hour acrophase shift.



Date/Time	Event	Effectiveness FAST	Effectiveness Clone	Resevoir Level FAST	Resevoir Level Clone
Jan08_0300	wake	890	890	2751	2750
Jan08_0301	wakeInertia	840	840	2751	2750
Jan08_1900	sleep_CircShift_starts	923	923	2271	2270
Jan10_300	wake	892	891	2742	2742
Jan10_0301	wakeInertia	842	842	2742	2741
Jan10_1900	sleep	906	906	2262	2262
Jan12_0300	wake	932	933	2793	2793
Jan12_0301	wakeInertia	884	884	2793	2793
Jan12_1900	sleep	891	885	2313	2313
Jan13_0659	circ_shift_ends	967	968	2699	2701
Jan16_300	wake	974	976	2862	2866
Jan16_1900	sleep	897	898	2382	2386

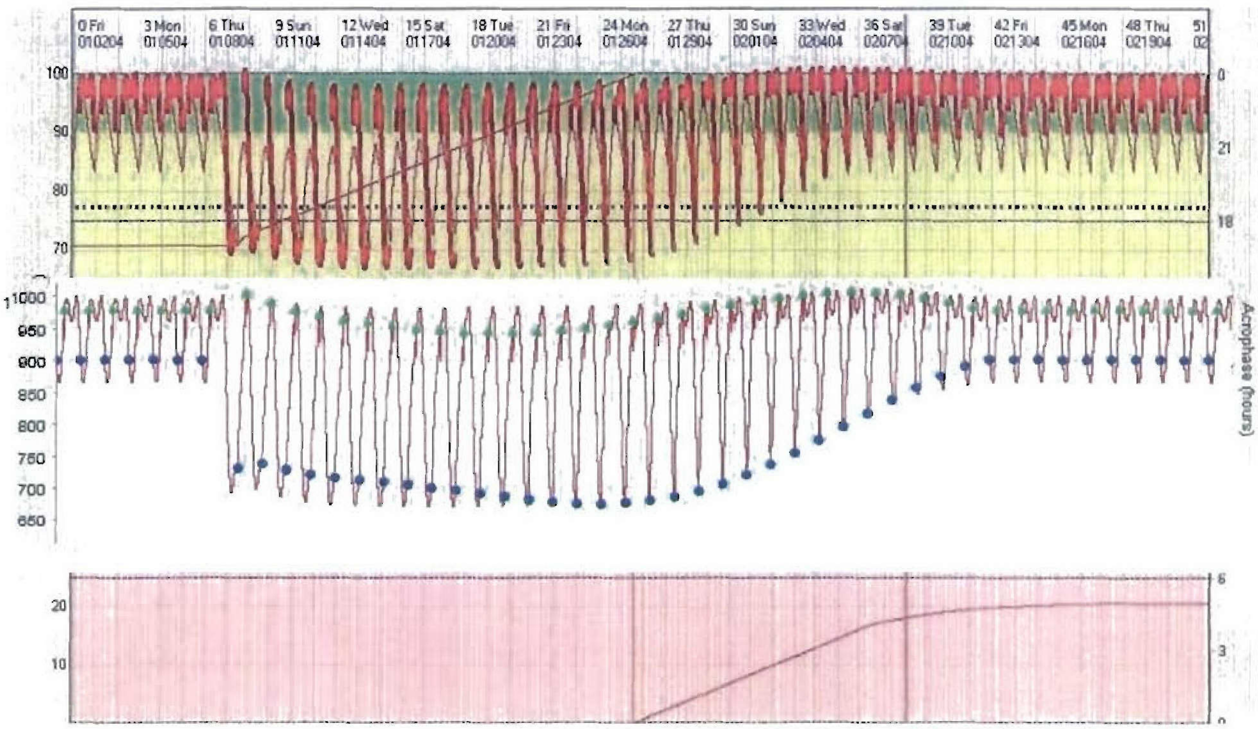
Table 3 is a 3-hour negative shift in bed time (sleeping from 1900 to 0300). This change causes a 3-hour advance in the location of circadian peak to reflect optimal placement. Negative acrophase shifts are more difficult for the brain and body to achieve than positive ones. The phase advance is implemented as a shift-rate adjustment of 1.5 days per hour of change. So for this kind of shift, 4.5 days is needed to re-align the circadian at its optimal position. Note that in this schedule sleep is entirely at night, because the clone assumes that it is dark at 1900. FAST appears to make a similar assumption. Therefore, it is not surprising that the FAST and clone numbers match more closely in Table 3 than in Table 2.

One interesting observation from Table 3 is that the circadian shift for the clone seems to end sooner than FAST , as shown by row Jan13_0659, which has the event "circ_shift_ends", to indicate that the target acrophase has been reached *from the clone's perspective*. In FAST one can plot an acrophase function (which appears below the effectiveness function) along with the effectiveness function, to observe circadian-shifting dynamics. For Table 3, the schedule acrophase starts at 1700 and then shifts to 1400. A vertical red-line (the FAST "cursor") has been

placed at the Jan13_0659 time point in the FAST output. As can be seen, FAST's acrophase function has not yet flattened out. Despite this, the actual fit of clone to FAST is close for this row. More on both observations in the following tables.

Table 4: 12hoursPhase shift

Baseline schedule migrates to 12 HOURS PLUS for 1000-1800 sleep-wake times and an asymptotic acrophase of 0500.



Date/Time	Event	Effectiveness FAST	Effectiveness Clone	Resevoir Level FAST	Resevoir Level Clone
Jan09_1000	sleep_CircShift_starts	731	731	2040	2040
Jan09_1800	wake	1004	1005	2547	2548
Jan12_1000	sleep	722	722	2036	2038
Jan12_1800	wake	972	972	2515	2518
Jan12_1801	wakeInertia	930	930	2515	2518
Jan17_1000	sleep	702	701	2074	2078
Jan17_1800	wake	948	948	2566	2571
Jan17_1801	wakeInertia	902	902	2566	2570
Jan27_1000	sleep	683	687	2209	2219

Jan27_1800	wake	971	975	2706	2717
Jan27_1801	wakeInertia	925	929	2706	2717
Feb09_0200	circShift_ends	974	963	2640	2640
Feb14_1000	sleep	895	902	2400	2400
Feb14_1800	wake	985	981	2880	2880

Table 4 shows a schedule that is shifted 12 hours in phase, so that the individual would be going to sleep at 1000 and waking at 1800. It takes a *long* time to adjust to this schedule, given that the amount of shift needed is large and 100% of the sleep is in the daytime. These factors lead to 2.6 days per hour of needed shift, or about 31 days of shifting. While a positive 12 hour shift is *mathematically* indistinguishable from a negative 12 hour shift, the nervous system is more readily inclined toward positive shifts, so both FAST and clone choose the easier direction to shift (but both will also choose a shorter negative shift vs. a longer positive shift).

Both FAST and clone shift the circadian rhythm from its starting acrophase of 1700 to a "later" 0500 target. (That is, there is a "wrapping around" effect on the FAST acrophase function as it passes through midnight at the top of the right-hand y axis, to reach its 0500 target, which is underneath it on that axis. Note also that the clone inset occludes the middle part of the FAST output in the figure). Again the FAST cursor has been placed at the time at which the *clone* acrophase shifting has stopped (Feb9_0200), and again the FAST acrophase function continues to show a small amount of shifting after that point.

Concerning this last difference, we have found out (from both the patent site and a discussion with Dr. Hursh) that FAST has a gradual deceleration dynamic as it approaches its new circadian acrophase target. This dynamic is completely independent of the other shifting rules (*for example*, the direction-of-shift and sleeping-in-the-daytime rules).[5]

---

5. This extra dynamic is designed to attain a new circadian acrophase very gradually and is detailed in patent point [0172], [0155-0156]. Essentially, the normal shift rate (as

This schedule is also interesting in theoretical terms because the individual gets an excellent 8 hours of sleep during each sleep period (at least we can program the individual to do so for both FAST and clone). However, there is a dramatic effect that is readily visible to the eye and lasts for the better part of a month. Both FAST and clone predict high amplitude swings in a person's effectiveness throughout the day as a result of switching to this schedule. One might think this is purely a circadian effect, but the reservoir is also impacted. Even quite a ways into the schedule (say midway at Jan 20) the reservoir level is around 2600 *upon awakening*, and this seems low relative to Table 1 (normal schedule) where the reservoir level upon awakening is 2880 (its maximum value). Clearly, the circadian rhythm, by its long-standing, non-optimal placement, is affecting reservoir accumulation. Why is this?

Recall that the circadian rhythm normally "picks up the slack" for the reservoir, being on the upswing while the reservoir is depleting (or the person is awake) and being on the downswing while the reservoir is filling up (or the person sleeps). The 12-hour phase shift reverses this relationship for a long period, so that the circadian may be amplifying the intensity of sleep and attenuating alertness while the individual is awake. The higher sleep intensity may explain the steeper accumulation slope during sleeping; however, intense sleeping also causes a "down-

---

determined by other rules) gets multiplied by the remaining difference needed to obtain the target, but this only goes into effect when the circadian rhythm attains a 1.0-hour or closer distance from its target (so "remaining differences" to the goal are fractional values). Under this scheme, the closer the current phase is to the target phase, the slower the shift rate; however, the target is still reached with perhaps a day or two delay. The clone has no extra gradual slowing, because we wish to study this mechanism's impact on future schedules. However, for the schedules investigated here, gradual slowing has only a minimal effect. We note, in passing, that both FAST and the clone shift to a new circadian target using a micro-shift every minute (i.e. the amount to shift in a minute is: [amount of shift expected in the day] / [1440, the number of minutes in a day]). This may already be considered a "gradual" shifting mechanism by some definitions.

regulation" in the reservoir capacity.[6] SAFTE presumes that high sleep intensities are avoided by the nervous system, just as any other kind of intense stimulus would be. By lowering the reservoir capacity, sleep intensity is also lowered (sleep intensity is a function of the distance of the current reservoir-level from the *current* reservoir capacity). A lower reservoir capacity seems to lead to lower reservoir levels at waking, regardless of how intensely you sleep[7]. We will observe reservoir-capacity regulation in more detail in the next schedules.

On the other hand, while the individual is awake, the reservoir and the circadian are *both going down*. So instead of maintaining effectiveness at or above 90% as in a normal schedule (where the circadian rhythm compensates the reservoir), effectiveness now troughs below 70%.

One final comment should be made about sleep intensity. Sleep intensity can become negative, allowing the reservoir to *deplete while the individual is sleeping*. This interesting occurrence was discovered by trying an alternate method to "catch" the 12-hour phase shift schedule. Given that you had to switch to this schedule, a rational tactic would be to stay up all night and go to bed at 1000 the following morning (*i.e.*, one incurs greater sleep debt than normal). However, another tactic would be to go to bed at 2200 and wake up at 1800 the next day (*i.e.*, obtain a sleep glut,

---

6. One can clone the clone to get plots of other theoretical constructs beside the overall effectiveness prediction, so this observation is based on a plot of "reservoir capacity" as a function of time.

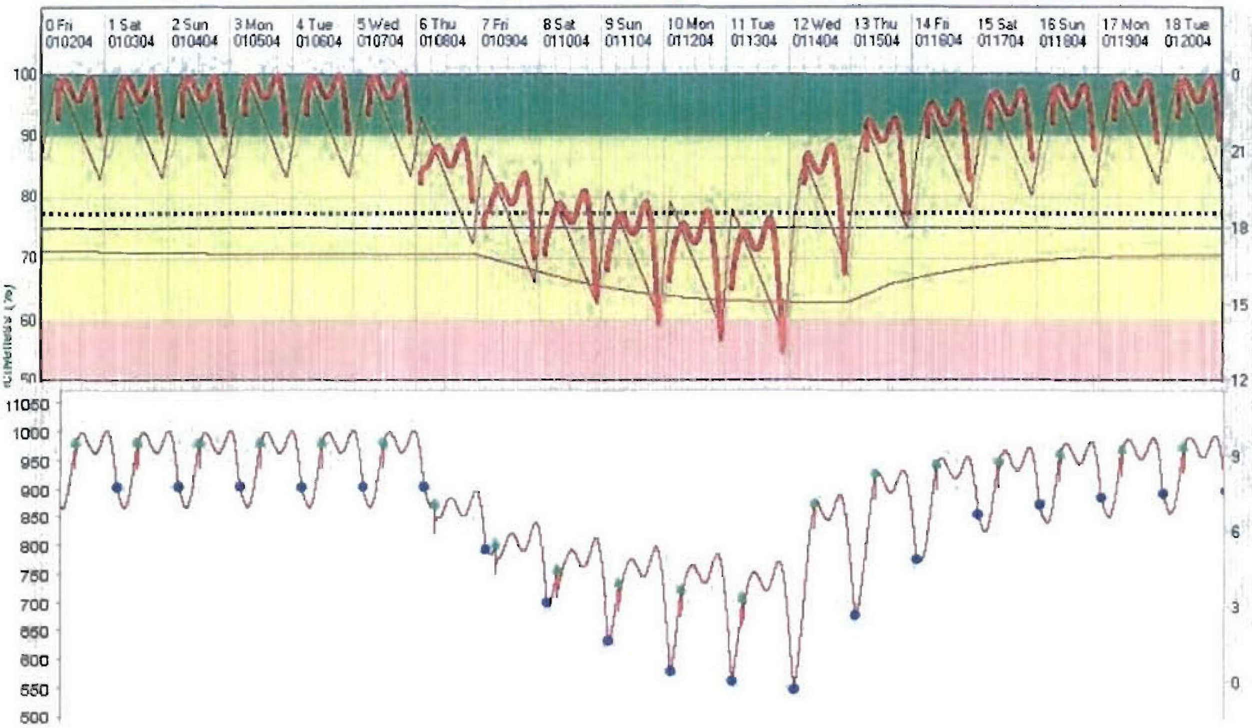
7. The effect of this schedule on reservoir capacity was smaller than the effect on (the highest level reached in) the reservoir. This can be seen by plotting reservoir level as a function of time. That is, given the reservoir capacity is "down-regulated" to be at 95% of its maximum, which was the finding here, does not mean that the individual will ever reach that reduced-capacity reservoir limit. In fact, only 88% of the reservoir's 100% natural capacity is ever reached (as can be seen by inspecting the reservoir level just before waking), and not the 95% that might have been expected from the new reservoir capacity setting. This interesting effect continues until the circadian position becomes more optimal and was also found for our other schedules, where down regulation was observed.

before starting the schedule).[8] With this bizarre strategy both FAST and the clone will show the reservoir depleting at the end of the extra-long sleep period as the sleep debt becomes zero and sleep propensity goes negative (sleepIntensity = sleepDebt + sleepPropensity). This occurs because neither implementation imposes a minimum sleep intensity value of zero (even though both implementations impose a maximum sleep intensity of 3.4 as part of the SAFTE specification, at least at some parts of the model). Negative sleep intensities may be theoretically interesting (*for example*, as a mechanism for "involuntary waking"), but we will defer that issue to the SAFTE scientists and engineers. Certainly imposing a minimum value of zero on sleep intensity is an easy option to adopt in future implementations, and this will probably not impact previous schedules studied (*i.e.*, predicted) by SAFTE.

8. We explored this option to assess the effects of the greater sleep debt incurred for the more "rational" option. This entering sleep debt had only minimal effect on the circadian "sleep intensity" effects noted for this schedule. That is when you enter the schedule with a sleep glut (and constrain sleep intensity to have a minimum of zero), the down-regulation of the reservoir still happens.

Table 5: 4hours_restrictedSleep

Baseline schedule 2200-0600 goes to 4 HOUR sleep restriction 2200-0200 sleep-wake, followed by recovery to the baseline schedule.



Date/Time	Event	Effectiveness FAST	Effectiveness Clone	Resevoir Level FAST	Resevoir Level Clone
Jan08_0200	wake	872	872	2686	2685
Jan08_0201	wakeInertia	822	821	2685	2684
Jan08_2200	sleep_CircShift_starts	792	792	2086	2085
Jan11_0200	wake	732	733	2334	2337
Jan11_0201	wakeInertia	681	683	2333	2336
Jan11_2159	circShift_ends	592	577	1734	1737
Jan14_0600	wake	872	875	2479	2485
Jan14_0601	wakeInertia	824	827	2479	2485
Jan14_2200	sleep_CircShift_starts	674	675	1999	2005
Jan15_0600	wake	924	926	2635	2639
Jan16_0600	wake	946	943	2733	2739
Jan16_2159	circShift_ends	827	854	2253	2259
Jan17_0600	wake	959	949	2792	2792
Jan18_2200	sleep	878	883	2348	2344
Jan19_0600	wake	973	969	2849	2845
Jan20_0600	wake	976	974	2861	2858

In Table 5 we see a sleep restriction schedule in which an individual changes from the baseline schedule to a 4-hour, 2200-0200 sleep period for 6 days followed by a recovery period (*i.e.*, back to baseline schedule). If you are forced to sleep only half of your normal amount, your sleep will be fairly intense and recovery from restricted sleep schedules will take some time: the person will not have recovered fully to baseline levels after 3 days. Downward regulation of reservoir capacity was implemented in SAFTE to accommodate slow recovery from a restricted sleep schedule. Recovery is a slow process because it takes time for a reservoir that has been "down-regulating" to "up-regulate" again.

The down- and up-regulation rates are not symmetrical. The down-rate depends on how much your sleep has been restricted, which determines how high your sleep intensity is when you sleep, which finally determines how fast you down-regulate to alleviate the intensity. When regulating up, one usually recovers using 8 hour sleep periods, and it is an open question as to whether one would recover more quickly using greater than 8 hour sleep periods during recovery (although the SAFTE model would predict this).

The table values for periods when peak effectiveness is declining from restricted sleep (down-regulation period) and the table values for periods when the individual recovers (up-regulation period) are mostly within 1% error. The two exceptional cases are when the circadian shifting ends (for the clone). Here, there is a 2.5% to 3.2% deviation of the clone from FAST in predicted effectiveness. Note that the deviations on the *reservoir levels* at these points are small and also in line with the more-closely agreeing effectiveness cases. This indicates that the large deviations *in effectiveness* observed at these points are due to circadian alignment differences between FAST and clone at these points. Recall FAST has extra circadian-shifting dynamics, a gradual slowing, and so will not reach its target by the time the clone does. In this particular schedule, that observation can be used to explain the most deviant cells. However recall that, in the schedule

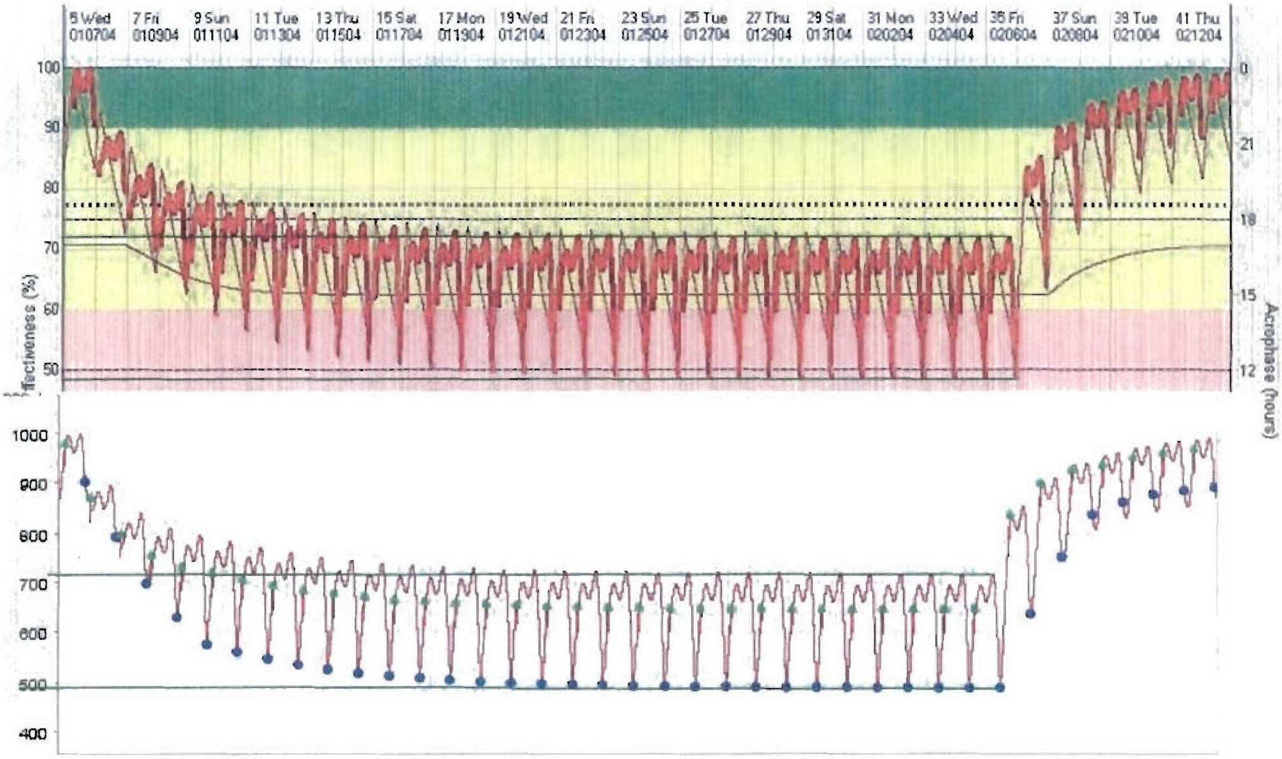
for Table 3, both the effectiveness levels and the reservoir levels matched much more closely at the point where the circadian shift ended for the clone. Also Table 4 (the 12 hour phase shift) didn't show as bad deviations for those schedule points as here. Why the difference for this schedule?

Circadian rhythms are not linear functions. Sometimes they are relatively constant in slope for about 3 hours, and sometimes they are relatively variable (slope changes rapidly). Note that for Table 3, the circadian shifting stops for the clone right *after* a relatively flat portion of the effectiveness function (about 4 hours after waking). It may be that in such cases FAST's extra slowing relative to the clone's would not be as noticeable as at some other times. For example, when the circadian rhythm was increasing at a constant rate for a long period (thus cancelling the reservoir depletion (a simple linear function downward), the slowing dynamics in FAST may not matter, but were the circadian to be varying greatly in slope at about the time the clone finished shifting, the extra FAST dynamic could become more noticeable.

While the above "casual argument" is not a proof, it can be verified, at least partially, by temporarily implementing the FAST gradual-slowing dynamics in the clone. When this is done for Schedule 5, the two circadian-shift ending times (the old-clone reference points) show predicted effectiveness values of 594 and 839, in temporal order, with reservoir dynamics unaffected as expected. The first value is very close to what FAST predicted for that point, and the error of the second value has been more than halved. Finally, adapting FAST gradual slowing dynamics in the clone for the schedule in Table 3 *still* produces a *968* effectiveness value, so adding the gradual-slowing dynamic retains the non-difference that was observed before for Table 3. In summary, these observations tend to support the casual argument given above.

Table 6: 4hoursRestricted_asyptotic

Baseline schedule 2200-0600 goes to 4 HOUR sleep restriction 2200-0200 TILL ASYMPTOTE, then recovery to the baseline schedule.



Date/Time	Event	Effectiveness FAST	Effectiveness Clone	Resevoir Level FAST	Resevoir Level Clone
Feb06_0200	wake	647	648	2095	2098
Feb06_0201	wakeInertia	596	597	2095	2097
Feb06_2200	sleep	486	488	1495	1498
Feb07_0600	wake	837	838	2379	2382
Feb07_0601	wakeInertia	788	789	2379	2382
Feb07_2200	sleep_CircShift_starts	635	637	1899	1902
Feb08_0600	wake	901	902	2570	2572
Feb08_0601	wakeInertia	853	854	2569	2572
Feb08_2200	sleep	745	751	2090	2092
Feb09_2159	circShift_ends	812	839	2212	2216
Feb10_0600	wake	950	938	2766	2764
Feb10_0601	wakeInertia	903	891	2766	2763
Feb10_2200	sleep	850	862	2286	2284
Feb13_0600	wake	974	971	2855	2851
Feb13_2200	sleep	891	892	2375	2371
Feb15_0600	wake	979	978	2871	2870
Feb15_2200	sleep	898	899	2391	2390
Feb16_0600	wake	980	979	2875	2874

Feb16_2200	sleep	900	900	2395	2394
Feb17_0600	wake	981	981	2879	2877
Feb17_2200	sleep	901	901	2398	2397

Table 6 is a variation on Table 5, essentially bringing the sleep-restriction schedule out to asymptote and then letting recovery happen. The result of long-term sleep restriction is asymptotic performance that peaks a little over 70% and troughs at about 50% (see also the green guide lines in the figure). The tabled values concentrate on the asymptotic and recovery portions only, and the match between FAST and clone is very good.

Table 7: 4hourRestricted_1hourNap_asymptotic

Schedule in Table 6 MODIFIED to include 1 HOUR NAP from 1400-1500. Table of comparisons are not shown because asymptotic acrophase predictions differ between FAST (about 19.5) and CLONE (15.0).

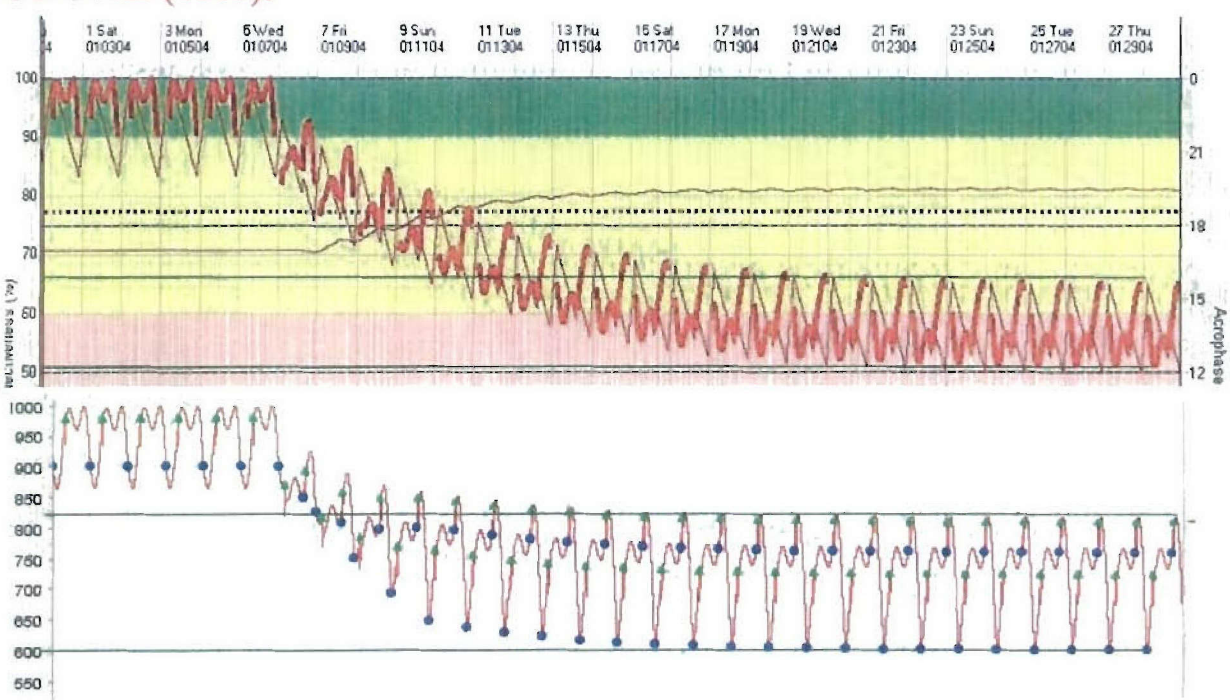


Table 7 modifies Table 6 by adding a one-hour nap at 1400 to the asymptotic 4-hour sleep restriction schedule. We decided to investigate schedules with naps because Hursh *et al.* (2004) did not discuss how the computation of circadian acrophase is affected by the presence of naps.

The clone uses an immediate tracking algorithm for computing acrophase. This algorithm directly observes the last midpoint of the waking period and adds 3 hours to it. The trick in this algorithm is how to define what the waking hours are when naps are present. We selected a policy in which naps could be considered "noise" with respect to determining target acrophase, so long as a suitable anchor sleep of reasonable duration could be identified at any part of the schedule. This schedule knowledge is embodied in some parameters (xml attributes for the "time0" element) that determines how much sleeping time (not necessarily sleep) has to accumulate before the sleep period is deemed over (i.e. a minSleepDuration attribute) and how much waking time (that can include nap time) has to accumulate before the waking period is deemed over (i.e. a minWakeDuration attribute). This policy was suggested by Minors & Waterhouse (1992). Given this policy, the acrophase for the schedule in Table 7, with respect to the clone, would be the same as in Table 6, which is 1500 hours (*i.e.*, the midpoint of the waking hours 0200 to 2200, plus 3 hours, where the nap is just considered part of the waking hours).

When we compared clone output to FAST, the difference was striking. Clone predicted that the addition of a nap (or an extra 55 min of sleep per day) would raise both the peak and trough effectiveness values to 82% and 60%, respectively. This result is appreciably better than the 70% and 50% values noted for the schedule without the nap. FAST predicted the addition of the nap would *depress* the peak to be below 70%, whereas the trough would only increase a little to just above 50% (see green guidelines in the Table figures for both implementations). Our initial reaction to FAST predicting degraded performance with added sleep was alarm, as we do not think many fatigue researchers would make this prediction.

FAST predictions also vary from clone predictions with respect to the acrophase that is sought asymptotically (see the red acrophase function in the FAST part of the figure). FAST predicts an acrophase target of about 1930, which is substantially delayed compared to where the clone wants to put the acrophase (1500). If we alter the clone so that it is *forced* to seek a 1930 acrophase asymptotically, the clone predictions correspond closely to the FAST results in Table 7. Conversely if we *force* FAST to seek an acrophase of 1500, instead of its desired 1930 target, the FAST predictions correspond closely to the clone results in Table 7. Therefore, the difference in FAST and clone predictions in Table 7 may be viewed as *entirely* owing to the *different* acrophase targets sought by each implementation of SAFTE.

Our next concern was whether or not FAST was behaving according to the SAFTE specification (there was insufficient detail in Hursh *et al.*, 2004, to determine this). After a discussion with Dr. Hursh and subsequent review at the patent website, we learned the precise algorithm by which FAST computes the acrophase in the presence of naps (see patent points: [0132] to [0141]). This is essentially a running average algorithm that uses information from the last 3 wake periods, but differentially weights the last wake period the most. We applied the algorithm and analytically determined that FAST *was* abiding by the SAFTE specification. The computation and detailed explanation of the effects seen in Table 7 for the FAST graphic are given in Appendix A.

Table 8: 48hourSustainedWake

Sustained awake (48 hours) starting at normal wake up time, followed by sleep for 16 hours, leading to secondary wake up at normal bedtime (2200), followed by a sustained wake of 24 hours leading to a resumption of normal schedule (2200-0600). Clone acrophase predictions for the first sleep (after 48 hour sustained wake) and second sleep (after 24 hours) are annotated in the clone

part of the figure. Quantitative comparisons FAST to clone are not shown because acrophase predictions differ between FAST and CLONE (*i.e.*, acrophase "delays" after first sustained wake for FAST but acrophase "advances" after this wake for clone).

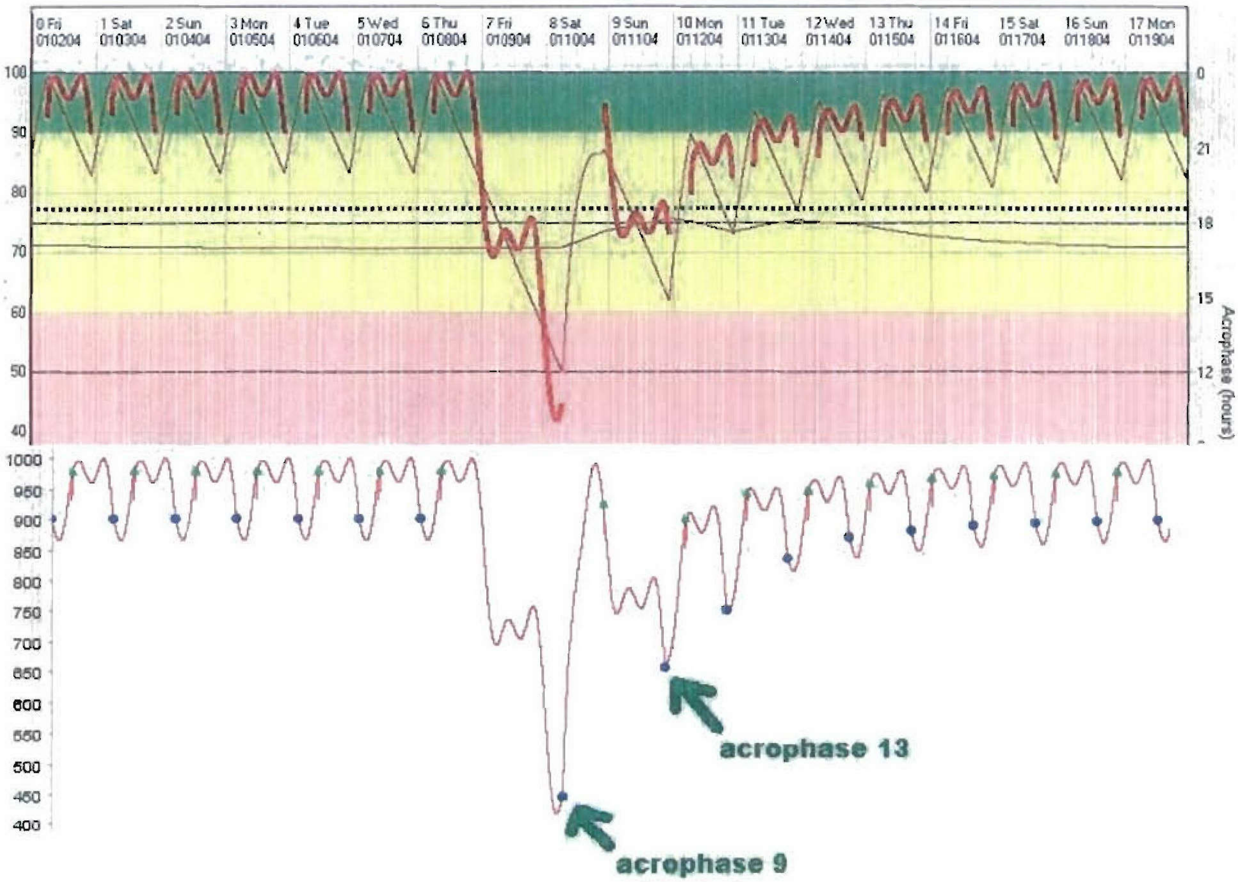
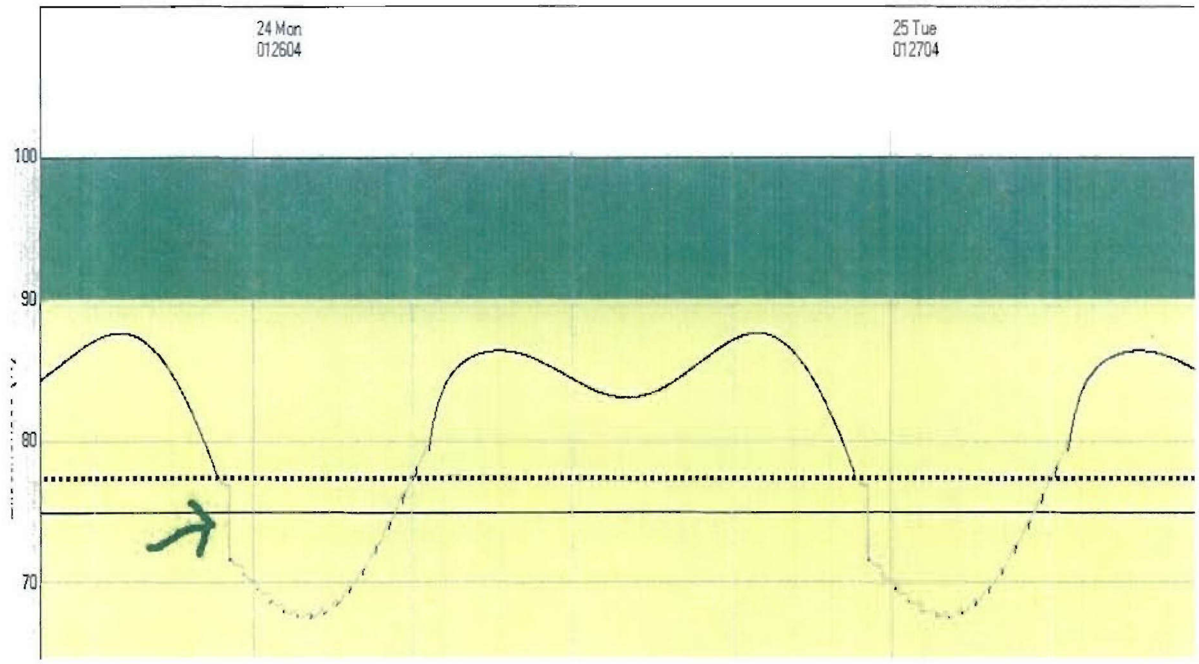


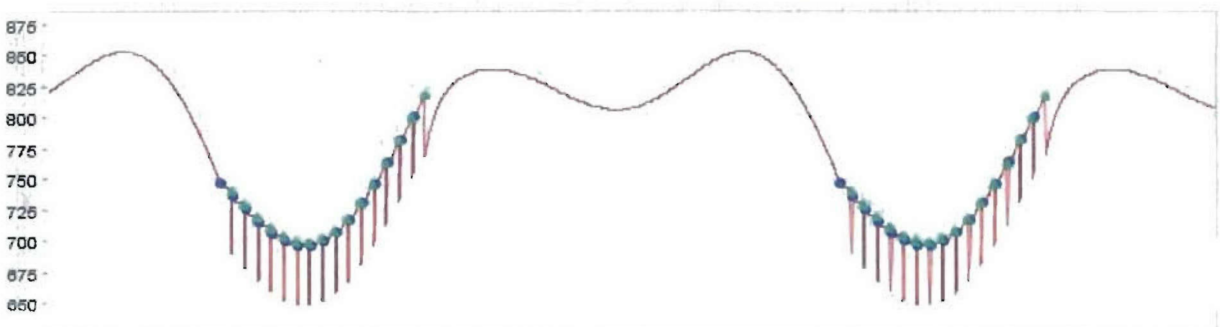
Table 8 involves 2 long wake periods. The first wake period ends at the normal wake-up time after a 48 hour sustained wake period. The second sustained wake period is 24 hours long and follows an abnormally long recovery sleep (16 hours that follows the 48-hour sustained wake period). The second sustained wake period gets the individual back to the baseline schedule. This schedule may not be operationally realistic but may allow us to study FAST and clone predictions for abnormally long wake periods, or for an acute sleep deprivation schedule. There is good agreement up to the first sleep after the 48 hour wake. However, at the first sleep FAST seeks an acrophase delay (later on the clock; higher on the right-hand y axis), whereas the clone seeks an advance (see the figure and the green arrows for clone acrophase targets at the onset of each sleep after a sustained wake).

We thought initially that the differences shown in Table 8 would be for the same reason as those shown in Table 7. However, applying Table 7's algorithm to Table 8's schedule, we found a new target of 0926, which is a phase advance close to the 0900 value that the clone predicts! We went back to the patent site and found a variance of the SAFTE algorithm for computing the midpoint of a waking period. This variance comes into effect when a waking period exceeds 24 hours (patent point [0144]). These extra long wake periods are handled as a special case, even though the algorithm for less long wake periods can be applied to these longer periods without mathematical anomaly. We defer the details of our verification of the special case to Appendix B. For now, we note that by following the variance, and *then* applying the equations for acrophase computation as in Table 7, a phase delay from 1700 local time to about 2100 is predicted by FAST, after the 48 hour wake. This is consistent with what the acrophase function is doing in FAST graphic for Table 8. After 24 hours awake, there was no special variance needed, and given the method from Table 7, a new acrophase target of 1024 was predicted. This seems consistent with where the acrophase function is heading after that waking period (*i.e.*, earlier on the clock). Eventually a return to the normal schedule puts the acrophase at 1700.

Table 9: fragmentedSleep

A near baseline schedule (2200-0555) with a sleep period "punctuated" by forced 5-minute wake ups at 30 min intervals. Acrophase is largely unaffected (approximately the same as baseline). The first set of table values are intermediate as verification comparisons. See Table 9A for the more final verification values along with the explanation for how that table differs from this one.





Date/Time	Event	Effectiveness FAST	Effectiveness Clone	Resevoir Level FAST	Resevoir Level Clone
Jan25_2225	firstWake	768	741	2067	1997
Jan26_0555	lastWake	801	819	2514	2444
Jan26_0830	firstPeak	864	839	2436	2366
Jan26_1815	secondPeak	876	853	2143	2073
Jan26_2200	initialSleep	772	747	2031	1961
Jan27_0100	sleepTrough	677	696	2235	2166

The last schedule that we investigated is a fragmented sleep schedule, implemented by waking the individual up for 5 minutes every 30 minutes during the normal 2200 – 0600 sleep/wake period (Tables 9 and 9a). Once awakened, the individual stays awake for 5 minutes before going back to sleep. Here we see FAST handling waking inertia differently from the clone. There is a large inertial effect on the *first* wake-up (see green arrow on the FAST part of the figure) which does not recover once the individual goes back to sleep. The clone does recover when the

individual goes back to sleep, because the clone's interpretation of the SAFTE specification is that "waking inertia" should not be expressed while sleeping. The result of this difference is an inertial "beard" for the clone, while FAST shows inertial spikes that resemble "stubble" (because a previous inertia has not recovered). One alarming factor is that the non-sleep portions of the curve (*i.e.*, effectiveness while awake) no longer match as well as before for *reservoir levels* (see values for "firstPeak" and "secondPeak"). This disparity indicates differences above and beyond just how inertia is being displayed for the FAST and clone implementations.

After some investigation, the following policy difference between clone and FAST was discovered for the 5-minute "sleep-latency" period. The sleep-latency period is the 5 minute period just after going to sleep. It is named so because reservoir replenishment starts only after 5 minutes into the sleeping period; a kind of warm up effect. The clone continues to "leak" from the reservoir during the sleep-latency period, while FAST holds the reservoir level constant. Under the clone's policy, the individual is losing 20 minutes of sleep-time every hour and draining the reservoir 20 minutes every hour. This follows from having two 5-minute awake periods plus two sleep latency periods during which the reservoir leaks. For FAST, there is also a sense in which the individual loses 20 minutes of sleep time, but the concomitant loss to the reservoir is halved compared to the clone.

Table 9a:fragmentedSleep_revised

For these table values, the clone adopts the policy (observed in FAST 1.0.26) to keep its reservoir level constant during the 5 minute "sleep latency" period. The policy for the clone depicted in Table 9 is for the reservoir to continue leaking during the 5 minute sleep latency.

		Effectiveness		Resevoir Level	
Date/Time	**Event**	**FAST**	**Clone**	**FAST**	**Clone**
Jan25_2225	firstWake	768	768	2067	2073
Jan26_0425	4thFromLast_wake	744	792	2452	2459
Jan26_0555	lastWake	801	847	2514	2519
Jan26_0830	firstPeak	864	867	2436	2442
Jan26_1815	secondPeak	876	879	2143	2149
Jan26_2200	initialSleep	772	773	2031	2037
Jan27_0100	sleepTrough	677	725	2235	2242
Jan27_0127	wakeInertial	676	677	2270	2278
Jan27_0426	wakeInertial2	740	742	2451	2458

A careful reading of the patent site (points [070] and [089]) suggests that the constant-reservoir policy during sleep latency could be more consistent with the SAFTE specification, even though the empirical finding supporting the "sleep fragmentation" mechanism in SAFTE (i.e., "12 awakenings per hour [...] equivalent to total sleep deprivation", Hursh 2004, pg. A47), does not, by itself, disambiguate policies. We implemented a non-leaking policy for sleep latency and re-did Table 9's numeric comparisons in Table 9a, adding extra row events to help diagnose the inertia-display differences.

Even though some systematic difference remains between clone and FAST in Table 9a, *all* of the reservoir-level readings are well within %1 error (unlike Table 9). Notice also that the non-

inertial events have noticeably divergent effectiveness values, such that FAST continues to express inertia while clone does not.[9]

Our best conjecture at this point is that clone/FAST differences showing up in Table 9a are probably due to a simple programming bug on the FAST end. This bug seems related to whether inertia should be "expressed" in the effectiveness predictions of an individual who goes to sleep shortly after waking up (at least this characterizes the difference between clone and FAST). If inertia is not so expressed, FAST output (i.e. its graphic) should look like the clone's and the remaining Table differences will likely go away.

DISCUSSION

All in all we were pleased with the amount of verification we demonstrated. Eight of nine schedules were adequately verified, in the sense of either showing good quantitative match between FAST and clone, or in the sense that an analytical discussion (with empirical verification) could explain the differences between FAST and clone. The ninth schedule, while still unresolved as a verification, seems consistent with a minor programming bug.

It is useful to break down the implementation differences between FAST and clone into categories defined by size and type of effect. Inconsequential differences between the two implementations were found from 1) different methods for computing circadian shifting dynamics from poor cues (percent-sleep-in-daytime), and 2) gradual slowing by FAST within a 1-hour distance of reaching the target acrophase.

---

9. However, the very first wake in the fragmented sleep period, which is also non-inertial (see also footnote 3), is an exception to the last observation.

Another category of implementation difference could be termed "schedule dependent", as with the policy of having the reservoir leak or hold a constant reservoir level during the sleep latency period. While we elected the latter for verification purposes, we note that the issue of which policy to implement is still open.[10] Moreover, as selection of one policy or the other will significantly impact predictions for schedules specifically involving heavily fragmented sleep (*i.e.*, different asymptotic reservoir-levels predicted), use of SAFTE/FAST in those circumstances may require a validation study to see if the no-leaking option is a more accurate reflection of sleep loss under those circumstances.[11]

The final category and perhaps the most important includes arbitrary implementation differences that can lead to divergent outcomes between the clone and FAST (at least we think that these implementation differences may be arbitrary). These differences could be serious to the extent the possible outcomes of the implementations are not equally "reasonable" with respect to the empirical literature. For Table 8, we have a non-obvious midpoint calculation for a sustained-wake period that exceeds 24 hours. This leads to an acrophase delay rather than advance for this

---

10. Dr. Miller comments: The SAFTE/FAST approach does not account for the standardized manner in which sleep is scored in research and clinical settings. Specifically, sleep latency occurs between the beginning of time in bed (TIB), when the room lights are turned off, and the appearance of (usually) the first epoch of stage 2 sleep. In practice, a sleep period in the SAFTE model is equated with TIB in the laboratory. However, latency consists nominally of 5 min of awake time after lights out, followed by 1 or 2 min of stage 1 sleep. Neither awake time nor stage 1 are viewed by the sleep research and clinical communities as being "sleep" in its restorative context. Thus, it would seem to make more sense that the reservoir continue to deplete during sleep latency than simply to pause in accumulation.

11. Given the "no-leaking" sleep latency policy was adopted/discovered *at the last schedule* the numbers for clone predictions are off (both effectiveness and reservoir level) in all but one of the preceding tables. For schedules where the maximum reservoir capacity *is not reached*, a no-leaking policy results in a value 2.5 reservoir units higher than the reservoir levels reported in those tables. For the baseline schedule, both policies lead to same result, in the clone, because the clone reaches its maximum reservoir capacity about five minutes before wake (while FAST tends to reach it more at the very end, a very slight difference). This means the extra 2.5 reservoir units (no-leak vs. leak) become "unexpressed" by the constraint of a maximum reservoir level.

situation. In the case of Table 8, the possible outcomes may not vary in "reasonableness". However, in the case of Table 7, this issue is difficult to dismiss. It seems odd that adding sleep should ever reduce mean effectiveness, but it does in FAST's implementation. Whether or not this remains in the SAFTE specification, we leave to stakeholders in the technology. However, now that we know different acrophase computations/implementations can lead to highly divergent effects (such as a decrease or increase in effectiveness from adding a nap), we would suggest more scrutiny be placed on validating the circadian shifting dynamics.

REFERENCES

Hursh, S. R., Redmond, D. P., Johnson, M. L., Thorne, D. R., Belenky, G., Balkin, T. J., Storm, W. F., Miller, J. C., & Eddy, D. R. (2004) Fatigue Models for Applied Research in Warfighting. *Aviation Space and Environmental Medicine*, 75(3), 44-53.

Minors, D. S., & Waterhouse, J. M. (1992) The Impact of Irregular Sleep-Wake Schedules on Circadian Rhythms and the Role of "Anchor" Sleep. In *Why We Nap: Evolution, Chronobiology, and Functions of Polyphasic and Ultrashort Sleep.* Stampi, C. (ed) Birkhäuser, Boston.

Defense Acquisition University (2004). Intermediate Systems Planning, Research, Development, and Engineering Training. Course offered online at: www.dau.mil

APPENDIX A

Formulas (patent point [0140]):
CurrentWeight = (Awake Duration of most recent awake period)/120 [minutes]

Weighted Average Goal Phase = [0.3333 * AH(1) + 0.6667 * AH(2) + CurrentWeight * AH(3)] / (CurrentWeight + 1)

Where AH(1) is waking hours for the first (most distant in past) "awake hours" interval.

Intermediate quantities:
Sleep intervals [2200 – 0200] , [1400 – 1500]
Wake intervals [1500-2200], [0200-1400]
7, 12, 7, 12, 7, 12, 7 ..... waking durations
1830, 0800, 1830, 0800 ..... wake interval midpoints
2130, 1100, 2130, 1100 .... Above midpoints + 3hours
(2130, 7), (1100 12), (2130, 7), (1100, 12) ..... Midpoints + 3hours yoked to durations

Recapitulation of general formula:
seek = [0.333(AH_{n-2}) + 0.667(AH_{n-1}) + currentWeight(AH_n)] / (currentWeight + 1)

seek1 defined by sleep durations 7, 12, 7, 12, ...
seek1 = [0.333(21.5) + 0.667(11.0) + currentWeight(21.5)] / (currentWeight + 1)
currentWeight = 420/120 = 3.5

seek1 = [0.333(21.5) + 0.667(11.0) + 3.5(21.5)] / (4.5)
seek1 = [7.166 + 7.337 + 75.25] / 4.5
seek1 = [89.753] / 4.5
seek1 = 19. 945

seek2 defined by sleep durations 12, 7, 12, 7, ...
currentWeight = 720/120 = 6.0
seek2 = [0.333(11.0) + 0.667(21.5) + 6.0(11.0)] / (currentWeight + 1)
seek2 = [3.7 + 14.3 + 66] / 7.0
seek2 = [84] / 7.0
seek2 = 12

assume circadian shifting starts, or is reconsidered, **after** you go to sleep (because that's when the "last waking period" becomes defined).

Time to shift after go to bed at 2200: 2200-1400 = 16 hours (this corresponds to seek1).

Time to shift after go to bed at 1400: 1400-2200 = 8 hours (this corresponds to seek2).

DETAILED DISCUSSION

This discussion details our reasoning for why FAST seeks the 19.5 (19:30) asymptote for the 4-hour sleep-restriction plus 1-hour nap schedule.

There are two waking periods in the asymptotic schedule for Table 7, so there are *only* 2 awake-hour (AHs) quantities, or two acrophase estimations, that can plug into the 3-slotted "running-average" equation which computes a target acrophase. This implies the running average equation will oscillate between the two possible values (and by implication so does the nervous-system of the person modeled), as half the time the AH from awake period 1 is leading (and ending) the equation, and the other half of time the AH from period 2 is leading (and ending).

Call one possible value for the equation, 'seek1' (defined by acrophase 21.5 or 21:30 leading) and the other 'seek2' (defined by acrophase 11:00 leading). As it turns out, seek1 = 19.9 (which is close to where FAST puts the acrophase asymptotically) and seek2 = 12.0 (very distant from where FAST puts the acrophase asymptotically).

So why does FAST put the acrophase asymptotically at 19.5 instead of *precisely* at 19.9 (you would need a transparent ruler held up to screen to see that it doesn't quite reach 19.9, but is actually midway between 18 and 21)?

Consider first why a *19.9-like value* may be expected in the FAST output (we will handle the slight-underestimation later). When the equation starts to indicate 19.9 should be the acrophase target, there are 16 subsequent hours in which to effect the requested shift. At the end of this time the equation indicates a 12.0 acrophase target. However, the 12.0 hour target only gets 8

hours of shifting time before acrophase target is redefined back to 19.9. So acrophase should crawl more toward 19.9 value with every succeeding day (i.e. each day is giving more shifting time to that target, and the shift direction, being positive, is easier, or faster, than the shift to a 12 hour target).

However, recall that the circadian shift speed also slows down as it gets closer to a target (see footnote 5 of this report). We believe this added dynamic causes the rate of shift to 19.9 to become so slow that the achieved asymptote will be somewhat lower than expected (i.e., 19.5 instead of 19.9). The slightly lower asymptote isn't embodied in the math above, but we feel it follows from the following straightforward argument.

Assume that you are already in the neighborhood of 19.9 and the running-average equation currently aims at 19.9. Then the shifting to target is extra-slow and getting continually slower. When the equation then aims at the 12.0 target, shifting to the target becomes "normal" but is relatively slow owing to the negative-shifts-are-slower rule of thumb. This scenario implies a homeostasis of sorts.

Initially, one climbs rapidly toward 19.9; however, within the 1-hour "event horizon" for the 19.9 target, slower temporal dynamics set in. At some point, the greater time and faster shift-rate for the 16-hour shift will be slowed by the gradual-slowing dynamic so as to accomplish just as much positive shift as the 12-hour negative shift.

In summary, the FAST asymptotic function will never become smooth (note the jaggedness in the figure). It is not a "damped" function in that sense. It is a half-damped function in the sense that the 16-hour shift segment eventually becomes equal in amplitude and opposite in direction

to the 8-hour shift segment, while the 8-hour shift component remains unchanged throughout the reaching of homeostasis.

APPENDIX B

Formulas (patent point [0140]):
CurrentWeight = (Awake Duration of last awake period)/120 [minutes]
Weighted Average Goal Phase = [0.3333 * AH(1) + 0.6667 * AH(2) + CurrentWeight * AH(3)] / (CurrentWeight + 1)
Where AH(1) is waking hours for the first (most past) "awake hours" interval.

Intervals:
Sleep interval [2200 – 0600]
Wake interval [600 – 2200]
Sleep interval [2200 – 0600]
Wake interval [0600 – 0600 – 0600] // up 48 hours;
Sleep interval [0600 – 2200] // sleep 16 hours to recover;
Wake interval [2200 – 2200] // up for 24 hours;
Sleep interval [2200 – 0600] // sleep 8 hours (and back to baseline sched);

Intermediate quantities:
16, 16, 16, 48, 24, 16, 16, ... waking durations
1700, 1700, 1700, 0600, 1000, 1700 ..... wake interval midpoints (literal computation)
2000, 2000, 2000, 0900, 1300, 2000.... Above midpoints + 3hours
[16, 2000] , [16, 2000], [16, 2000], [48, 0900], [24, 1300], [16, 2000] ..... durations yoked to midpoints + 3hours

Computations of acrophase target for given points in the schedule:

<u>No special variance applied:</u>

seek1 defined by awake hours: 16, 16, 48, going from most past to most recent:

application of formulas:
currentWeight = 2880/120 = 24
seek1 = [0.333(20) + 0.667(20) + currentWeight(9)] / (currentWeight + 1)

seek1 = [0.333(20) + 0.667(20) + 24(9)] / (25)
seek1 = [6.67 + 13.33 + 216] / 25
seek1 = [236] / 25
seek1 = 9.44 or 926 local time

seek2 defined by 16, 48, 24 , going from most past to most recent:

currentWeight = 1440/120 = 12
seek2 = [0.333(20) + 0.667(9) + 12(10)] / (currentWeight + 1)
seek2 = [6.67 + 6 + 120] / 13
seek2 = [132.667] / 13
seek2 = 10.2

seek3 defined by 48, 24, 16:
currentWeight = 960/120 = 8

seek3 = [0.333(9) + 0.667(10) + 8(16)] / (currentWeight + 1)
seek3 = [3 + 6.67+ 128]/9
seek3 = 137.67/9 = 15.3

Special variance applied:

Compute midpoint of 48 hour sustained wake via patent point [0144]:
"...For unusually long awake periods greater than 24 hours, the average hour of the last portion of the awake period less than 24 hrs is averaged with a time 12 hrs after the start of the awake period for each proceeding 24 hour period awake."

Wake interval [0600a, 0600b, 0600c]

// what to do with the END of the awake period according to [0144]
Average of 0600b+ 0600c is 1800 (visualize the average time from 0000a to 0000b, which is 1200, then phase delay that 6 hours)

// what to do with the start of the awake period according to [0144]
0600a+1200 = 1800

Average the two together: which is 1800

Add 3 hours to it for 2100 acrophase

Revised seek1 = [0.333(20) + 0.667(20) + 24(21)] / (25)
seek1 = [6.67 + 13.33 + 504] / 25 = 524/25 = 20.96

seek2 defined by 16, 48, 24:
currentWeight = 1440/120 = 12
seek2 = [0.333(20) + 0.667(21) + 12(10)] / (currentWeight + 1)
seek2 = [6.67 + 14+ 120] / 13
seek2 = [140.667] / 13
seek2 = 10.8

seek3 defined by 48, 24, 16:
currentWeight = 960/120 = 8
seek3 = [0.333(21) + 0.667(10) + 8(16)] / (currentWeight + 1)
seek3 = [7 + 6.67+ 128]/9

seek3= 141.667/9 = 15.7